THE SOCIETY OF NAVAL ARCHITECTS
AND MARINE ENGINEERS

**One World Trade Center, Suite 1369, New York, N.Y. 10048**

Paper presented at SCAHD'77
Computer-Aided Hull Surface Definition Symposium
Annapolis, Maryland
September 26–27, 1977

# B-Spline Curves and Surfaces for Ship Hull Definition

**David F. Rogers, Visitor**
**United States Naval Academy**
**Annapolis, Maryland 21401**



## Abstract

B-spline curves and surfaces and their suitability for ship hull surface definition are discussed. A comparison with other curve and surface definition methods, e.g., cubic splines, Bézier curves, parabolic blending is given. Their use both for ab initio curve and surface generation and for fitting existing offset data is discussed. Comparisons of the data storage requirements are made. The utilization of these concepts in an interactive design program implemented on an Evans and Sutherland Picture System supported by a PDP 11/45 and a Xynetics Model 1200 flat plotter are discussed. Initial efforts in generating numerical control tapes from the resulting data base and the manufacture of towing tank models are discussed. A demonstration of this program and the model manufacturing technique will be given.

## Introduction

During the past decade several methods of "fairing" and generating the lines for curves and/or for use in generating surface patches have been developed. Among these are piecewise cubic splines (1-5), piecewise cubic splines in tension (6,7), parabolic blending (1,8), piecewise circular arcs (10), and Bézier curves (1,9). All of these methods suffer to one extent or another from a number of difficulties. Among these are unwanted polynomial oscillations, limited continuity at the joins (data points), limited local control, inconvenient "handles" available for use in shaping the curve or surface by non-mathematical users, excessive computational requirements, excessive computer storage required to hold the curve, the necessity to break curves at sharp corners or knuckles and the necessity to represent curves in a piecewise manner. All of these methods have been used for ship line and surface

design and fitting. For example, piecewise circular arcs (10) are used in the "Autokon System", cubic splines and cubic splines in tension are used in the U.S. Navy CASDOS system (12), and Bézier curves are used in the Unisurf system (13), which has been used for ship hull design as well as auto-body design.

Recently a new type of curve has been developed which overcomes many of these difficulties. This is the B-spline curve (1,14). The concept is easily extended to Cartesian product surfaces.

B-spline curves and surfaces are capable of representing complex curves with a single mathematical formulation, continuity of high order can be maintained, excellent local control is possible, convenient and natural control handles are available, computational requirements for ab initio design are minimal, computer storage requirements to "hold" a curve are much reduced, and knuckles, hard chines and other sharp corners can be represented within one mathematical formula without resorting to breaking the curve. Before examining B-spline curves in detail a brief description of parabolically blended, piecewise cubic splines, and Bézier curves will be given.

**Parabolic Blended Curves**. A three dimensional parabolic blended curve (1,8,15) segment is defined by

$$\bar{P}(t) = \bar{R}(t) + t[\bar{Q}(t) - \bar{R}(t)] 0 < t < 1$$

where $\bar{P}(t)$ is a position vector with components $[\,x(t)\;\; y(t)\;\; z(t)\,]$ and $\bar{R}(t)$ and $\bar{Q}(t)$ are three dimensional parabolas in a local coordinate system through overlapping sets of three successive data points. $\bar{P}(t)$ is defined on the overlapping interval. The equation for the curve P(t) is cubic when expressed in terms of a basic Cartesian coordinate system and thus can exhibit an inflection point. Parabolic blended curves pass through all of the given data points.

Reference (15) shows that the parabolically blended segment can be written as

$$\bar{P}(t) = [\,t^3 \quad t^2 \quad t \quad 1\,]\,[\,A\,]\,[\,P_1 \quad P_2 \quad P_3 \quad P_4\,]^T$$

where

$$[\,A\,] = \begin{bmatrix} -1/2 & 3/2 & -3/2 & 1/2 \\ 1 & -5/2 & 2 & -1/2 \\ -1/2 & 0 & 1/2 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

and $P_1 \rightarrow P_4$ are three dimensional data points on the curve. Parabolic blended curves have second derivative continuity. As suggested by Reference (15) discontinuities in the first derivative, e.g., cusps, can be obtained by using points off of the blended curve. Further description is given in References (1), (8), and (15).

**Cubic Spline**. A three dimensional cubic spline segment is defined by

$$\bar{P}(t) = \sum_{i=1}^{4} \bar{B}_i t^{i-1}; \qquad t_l < t < t_2$$

where $\bar{P}(t)$ is the position vector with components $[\,x(t) \quad y(t) \quad z(t)\,]$ and $\bar{B}_i$ contains 12 constant scalar coefficients which are obtained by specifying four boundary condition vectors, each with three components. References (2)–(5), as well as Ref. (1) discuss the mathematical theory and implementation of cubic splines.

Cubic splines are piecewise continuous in position, slope, and second derivative. They pass through all data points given between the beginning and end of the spline. There is no unique spline for a given set of points because different end conditions can be used to create different curves, all of which pass through the given data points. One disadvantage of the cubic spline is that spurious wiggles can occur, especially if the data points are not equally spaced.

**Bézier Curves**. A three dimensional Bézier curve is defined by

$$\bar{P}(t) = \sum_{i=0}^{n} \bar{B}_i J_{n,i}(t); \qquad 0 < t < 1$$

where

$$J_{n,i}(t) = \frac{n!}{i!(n-i)!} t^i (1-t)^{n-i}$$

The shape of the curve is defined by a set of three dimensional vertices $\bar{B}_i$ which form an open polygon. The resulting curve is tangent to the first and last span of the polygon and passes through the first and last vertex. Otherwise the Bézier curve, the order being equal to the number of polygon vertices, is a smooth curve which can be easily controlled by positioning of the polygon vertices. The curve does not pass through the interior vertices. A typical Bézier curve and its defining polygon is shown in Fig. 1.

This curve was developed by P. E. Bézier for use in the design of Renault automobiles (13). It has proven to be a very practical means for defining complex curve shapes, and has been used by both engineers and artists. Further description of the Bézier curve is given in Refs. (1), (9) and (16).

## B-Spline Curves

Perhaps the most successful of the curves mentioned above is the Bézier curve. The B-spline curve had as its genesis the Bézier curve. Both Bézier and B-spline curves use a set of defining polygon points to provide controls for designing the resulting curves. From a mathematical point of view, a curve which is generated by using the vertices of a defining polygon is dependent on some interpolation or approximation scheme to establish the relationship between the curve and the polygon. This scheme is provided by the choice of a basis or weighting function. Bézier curves use a Bernstein basis or weighting function.

Two characteristics of the Bernstein basis, however, limit the flexibility of the resulting curves. First the number of specified polygon vertices fixes the order of the resulting polynomial which defines the curve. For example, a cubic curve must be defined by a polygon with four vertices and three spans. A polygon with six vertices will always produce a fifth-degree curve. The only way to reduce the order of the curve is to reduce the number of

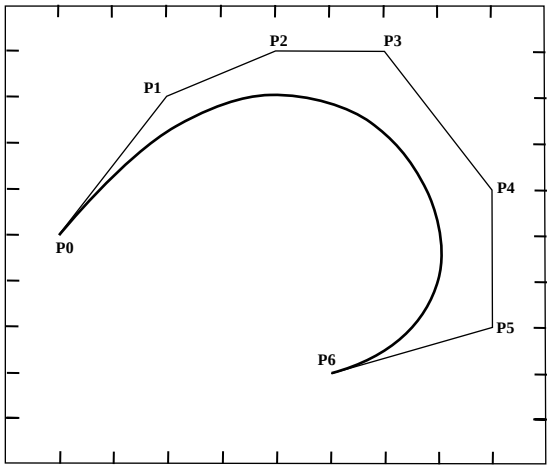


**Figure 1.** Bézier curve and defining polygon.

vertices, and conversely the only way to increase the order of the curve is to increase the number of vertices.

The second limiting characteristic is due to the global nature of the Bernstein basis. This means that the value of the weighting function $J_{n,i}(t)$ is nonzero for all parameter values over an entire span of the curve. Because any point on a Bézier curve is a result of weighting the values of all defining vertices, a change in one vertex is felt throughout the entire span. Practically, this eliminates the ability to produce a local change within a span. This lack of local span control can be detrimental in some applications.

There is another basis, called the B-spline basis, which contains the Bernstein basis as a special case. This basis is generally nonglobal. The nonglobal behavior of B-spline curves is due to the fact that each vertex $\bar{P}$ is associated with a unique basis function. Thus, each vertex affects the shape of a curve only over a range of parameter values where its associated basis function is nonzero. The B-spline basis also allows the order of the resulting curve to be changed without changing the number of defining polygon vertices. The theory for B-splines was first suggested by Schoenberg (17). A recursive definition useful for numerical computation was published by Cox (18) and by de Boor (19). Reisenfeld (15) originally applied the B-spline basis to curve definition.

If we let $\bar{P}(t)$ be the position vectors along the curve, as a function of the parameter $t$, a curve generated using the B-spline basis is given by

$$\bar{P}(t) = \sum_{i=0}^{n} \bar{B}_i N_{i,k}(t)$$

where the $\bar{B}_i$ are the $n+1$ three dimensional defining polygon vertices.

For the ith normalized B-spline basis curve of order $k$, the weighting functions $N_{i,k}(t)$ are defined by the recursion formulas

$$N_{i,l}(t) = \begin{cases} 1 & \text{if } x_i < t < x_{i+l} \\ 0 & \text{otherwise} \end{cases}$$

and

$$N_{i,k}(t) = \frac{(t - x_i)N_{i,k-1}(t)}{x_{i+k-1} - x_i} + \frac{(x_{i+k} - t)N_{i+1,k-1}(t)}{x_{i+k} - x_{i+1}}$$

The values of $x_i$ are elements of a knot vector and the parameter $t$ varies from 0 to $t_{\max}$ along the curve $P(t)$.

An additional variable must be used for B-spline curves to account for their inherent added flexibility. This is achieved by use of a knot vector. A knot vector is simply a series of real integers $x_i$ such that $x_i < x_{i+l}$ for all $x_i$. Examples of knot vectors are $[\,0 \quad 1 \quad 2 \quad 3 \quad 4\,]$ and $[\,0 \quad 0 \quad 0 \quad 1 \quad 1 \quad 2 \quad 3 \quad 3 \quad 3\,]$. The values of $x_i$ are considered to be parametric knots. They can be used to indicate the range of the parameter $t$ used to generate a B-spline curve with $0 \le t \le t_{\max}$. For example, the knot vector $[\,0 \quad 1 \quad 2 \quad 3 \quad 4\,]$ indicates that the parameter $t$ varies from 0 to 4. The number of intermediate knot values depends on the number of nonzero spans in the defining polygon. A duplicate intermediate knot value indicates that a multiple vertex (span of zero length) occurs at a point, and an intermediate knot value in triplicate indicates three concurrent vertices (two zero-length spans), etc. The actual point on a B-spline curve which corresponds to the value of a parametric knot ($t = x_i$) is called a geometric knot. It is convenient to use evenly spaced knots with unit separation between noncoincident knots. This gives integer values for the components of the knot vector.

In addition to the knot vector values, the order of the curve must be specified. If the order $k$ equals the number of polygon vertices, and there are no multiple vertices, then a Bézier curve will be generated. As the order decreases, the curve produced lies closer to the defining polygon. When $k = 2$ the generated curve is a series of straight lines which are identical to the defining polygon. The order of the curve is reflected in the knot vector. Knots of multiplicity $k$ are used at both the beginning and the end of the knot vector. The maximum value of the knot vector is $a + k - 2$ where $a$ is the number of nonzero spans in the defining polygon. The number of integers in the knot vector is $n + 1 + k$. For example, consider a five-point polygon ($n + 1 = 5$) with no duplicate vertices. When there are no duplicate vertices, the parameter $t$ varies from 0 to $n - k + 2$ over the entire curve. For a third-order curve defined by five vertices, the value of $t_{\max} = 4 - 3 + 2 = 3$. The complete knot vector, using multiplicity of 3 at each end, is then given by $[\,0 \quad 0 \quad 0 \quad 1 \quad 2 \quad 3 \quad 3 \quad 3\,]$. A second order curve for the same defining polygon has a knot vector $[\,0 \quad 0 \quad 1 \quad 2 \quad 3 \quad 4 \quad 4\,]$ and a fifth-order curve (which corresponds to a Bézier curve) has a knot vector $[\,0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1 \quad 1 \quad 1 \quad 1\,]$. Now, if the third and fourth vertices are made to coincide so that a multiple knot occurs, the knot vector for the second order curve is $[\,0 \quad 0 \quad 1 \quad 2 \quad 2 \quad 3 \quad 3\,]$.

Because a B-spline curve is mathematically defined as a polynomial spline function of order $k$ (degree $k - 1$) and if smoothness is based on continuity of higher order derivatives, the order of the curve determines how "smooth" the curve is. For example, a fourth order

(third degree) curve is continuous in first and second derivative, as well as position, along the entire curve. Thus, a fourth order B-spline curve is analogous to a paecewise cubic spline curve.

Due to the flexibility of B-spline curves, different types of control "handles" can be used to change the shape of a curve. Control can be achieved by changing the integer order $k$ for $2 < k < n + 1$, by use of repeating vertices, or by changing the number and/or position of nonrepeating vertices in the defining polygon. These effects are illustrated in the following six figures.

Figure 2 shows three B-spline curves of different order each defined by the same four polygon vertices given by

$$\begin{bmatrix} 0 & 0 \\ 3 & 9 \\ 6 & 3 \\ 9 & 6 \end{bmatrix}$$

The second-order curve creates three straight lines between the four vertices, the fourth-order curve corresponds to the Bézier curve for the polygon set and the third-order curve

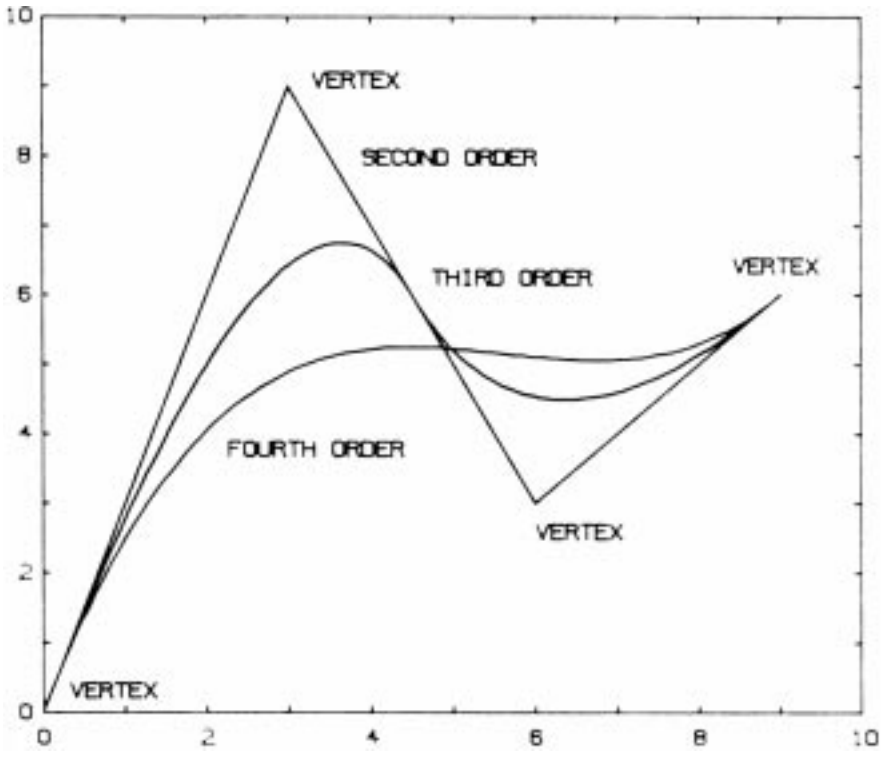


**Figure 2.** B-spline curves.

produces a looser curve between the two end points. Notice that all three curves have the same end slopes, determined by the slope of the first and last spans of the defining polygon. As the order of a curve increases, the resulting shape looks less like the defining polygon. Thus, increasing the order tightens the curve.

Figure 3 shows the effect of multiple vertices in the defining polygon. For each of the four curves shown, the order of the curve is equal to the number of vertices in the defining polygon. The lower curve in Fig. 3 is identical to the lower curve in Fig. 2, a fourth-order curve defined by four polygon vertices. The second curve in Fig. 3 is a fifth-order curve with a double vertex at [3 9]. The final seventh-order curve has a defining polygon given by

$$\begin{bmatrix} 0 & 0 \\ 3 & 9 \\ 3 & 9 \\ 3 & 9 \\ 3 & 9 \\ 6 & 3 \\ 9 & 6 \end{bmatrix}$$

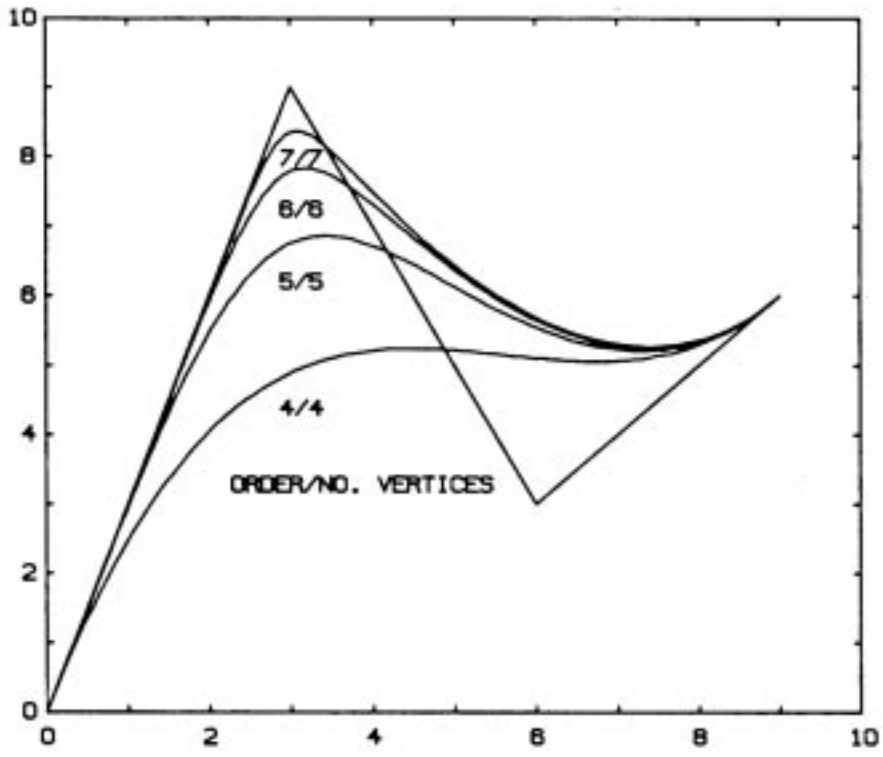


**Figure 3.** Multiple vertex B-spline curves.

i.e., four multiple vertices at [3 9]. This figure clearly shows how a curve can be pulled closer to a specific vertex position by use of multiple vertices while maintaining the same end slopes for each curve. On the other hand, decreasing the order pulls the curve closer to all polygon vertices.

In Fig. 4 the defining vertices are

$$\begin{bmatrix} 0 & 0 \\ 2 & 5 \\ 4 & 8 \\ 6 & 3 \\ 6 & 3 \\ 8 & 6 \\ 10 & 7 \end{bmatrix}$$

for each curve. That is, we keep a double vertex at the fourth element in the polygon. Here the curve is altered by changing the order of each curve, keeping the defining polygon

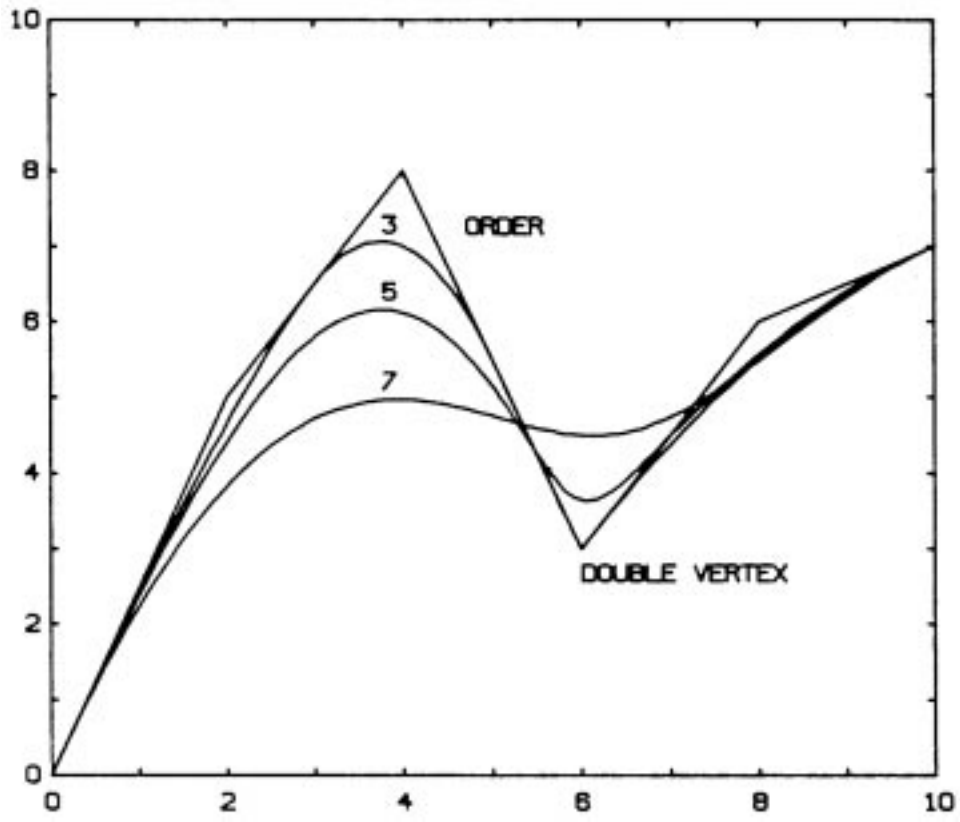


**Figure 4.** Multiple knot B-spline curves.

constant. The first curve is of order seven, equal to the number of vertices in the polygon. The second curve is of order five. This curve shape is closer to the polygon shape, especially near the double vertex. The third curve is of order 3. Notice that a "knuckle" occurs at the double vertex because the slope and curvature are discontinuous. A duplicate vertex is required to create a knuckle in a third-order curve. A triple vertex creates a knuckle in a fourth order curve, etc. This ability is a common requirement in ship design. Note also that even though both the slope and curvature are discontinuous, as is required by the existence of the knuckle, that a single uniform representation of the curve is still maintained. The curve is not broken or split at the knuckle as is required by other methods.

Figure 5 demonstrates how local changes can be made without affecting the entire shape of a curve. Each curve is a fifth-order curve, defined by a seven-point polygon with no multiple vertices. The only difference between each curve is that the fifth vertex is moved to a new position, as shown in the figure. It can be seen that the first part of each curve is unchanged. This behavior is a result of the nonglobal (local) nature of the B-spline basis.

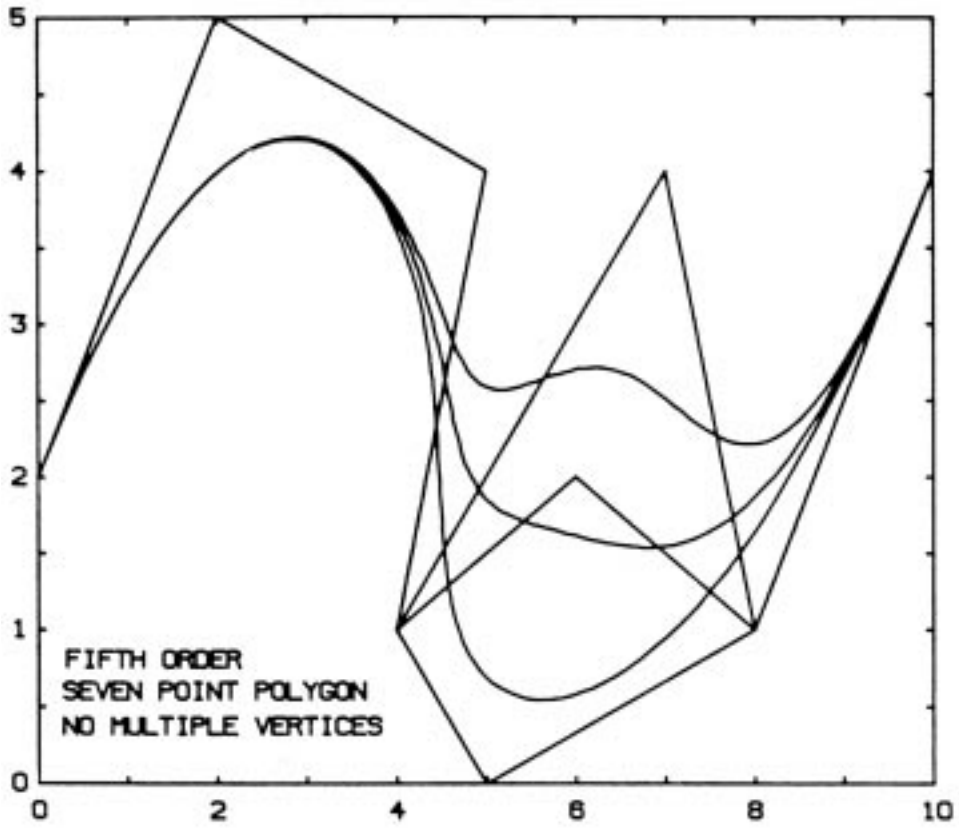


**Figure 5.** Local control of B-spline curves.

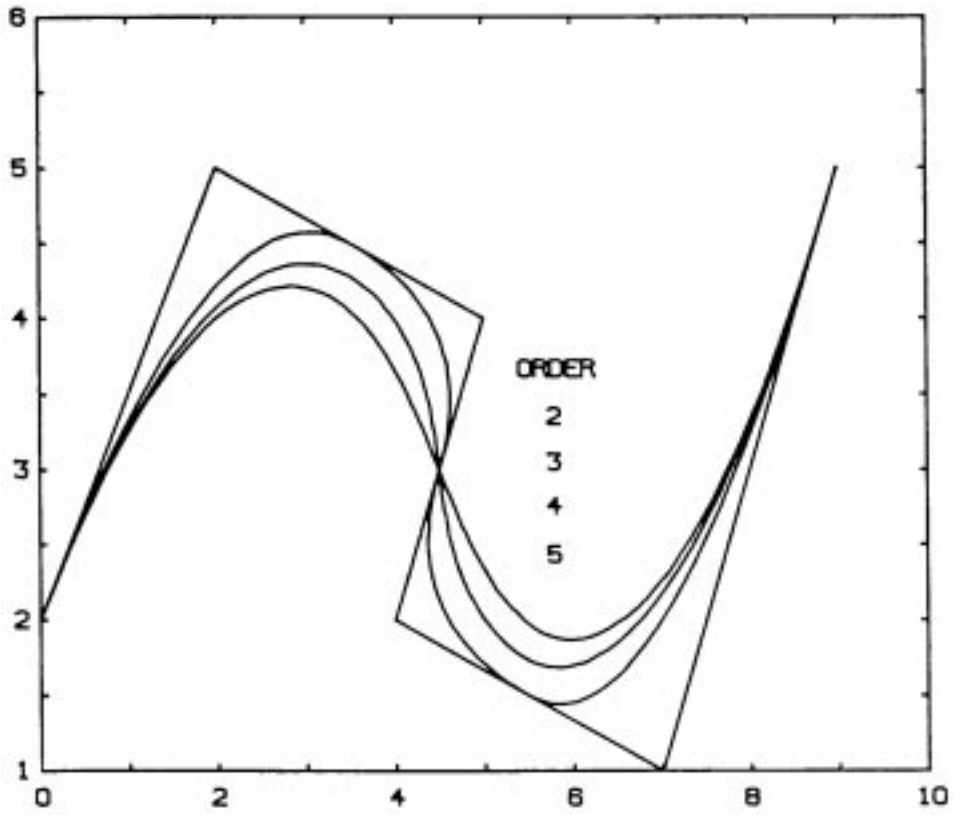


**Figure 6.** Varying order of B-spline curves.

Figure 6 shows curves of order 2 through 5, generated by a six-point polygon with no multiple vertices. The fourth order curve is a cubic spline.

The B-spline curve technique allows a cubic spline to be generated with three or more polygon vertices. Because there are five spans in this polygon, the fifth-order curve is the Bézier curve. The third-order curve may be of special interest because it is tangent to the midpoints of the internal polygon spans. This characteristic is also shown in Fig. 7, where a third-order curve is generated with an eight-point polygon.

## B-Spline Surfaces

Implementation of B-spline surfaces can take numerous forms. Perhaps the simplest is the Cartesian product surface given by

$$\bar{Q}(u,w) = \sum_{i=0}^{n} \sum_{j=0}^{m} \bar{B}_{i+1,j+1} N_{i,k}(u) M_{j,\ell}(w)$$

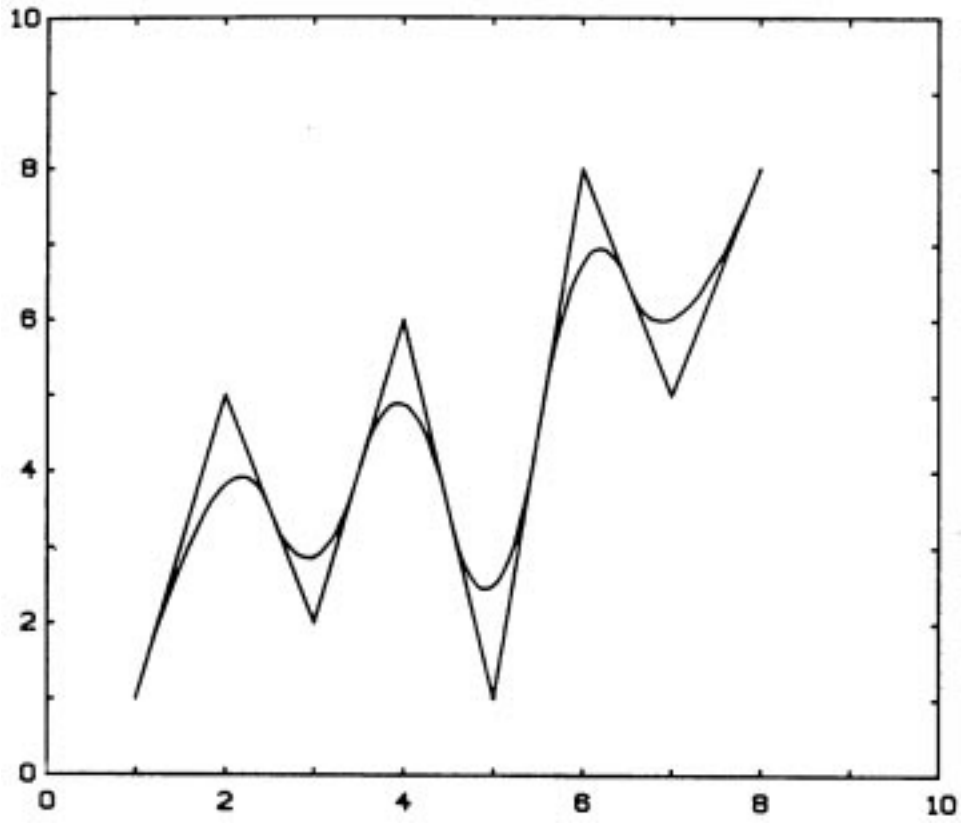


**Figure 7.** Third order B-spline curve.

where

$$N_{i,1}(u) = \begin{cases} 1 & \text{if } x_i \le u < x_{i+1} \\ 0 & \text{otherwise} \end{cases}$$

$$N_{i,k}(u) = \frac{(u - x_i)N_{i,k-1}(u)}{x_{i+k-1} - x_i} + \frac{(x_{i+k} - u)N_{i+1,k-1}(u)}{x_{i+k} - x_{i+1}}$$

and the $x_i$ are the elements of the $k$ knot vector.

Furthermore,

$$M_{j,1}(w) = \begin{cases} 1 & \text{if } y_j \le w < y_{j+1} \\ 0 & \text{otherwise} \end{cases}$$

$$M_{j,\ell}(w) = \frac{(w - y_j)M_{j,\ell-1}(w)}{y_{j+\ell-1} - y_j} + \frac{(y_{j+\ell} - w)M_{j+1,\ell-1}(w)}{y_{j+\ell} - y_{j+1}}$$

and the $y_j$ are the elements of the $\ell$ knot vector and $m$ and $n$ are one less than the number of vertices in the defining polygons in the orthogonal $u$- and $w$-directions respectively. The $B_{i+l,j+l}$ are the three dimensional position vectors of the defining polgonal surface.

As with B-spline curves, knot vectors with various degrees of multiplicity can be defined in either the $u$- or $w$-directions. However, the above formulation requires that each defining polygon in a given direction must have the same degree of multiplicity in a given direction. Alternate formulations allow various degrees of knot multiplicity in a given direction in order to increase the flexibility of local control. Further discussion of how B-spline curves can be used to form surface patches is given in Ref. (20).

## Fairing and Fitting

B-spline curves and surfaces as discussed above lend themselves extremely well to ab initio curve design. They can be easily modified using the polygon vertices as control handles. Their use and utility has generally been considered from this point of view. However, frequently a data base for a curve or surface exists, e.g., a set of digitized ships lines or offsets and it may be desirable to fit or fair a B-spline curve or surface to these data. Once an initial fit or fairing has been obtained the characteristics of the B-spline basis and of the control polygons can be used to modify the curve or surface to improve the fit or fairing. Very little work has been done in this area. What has been done has been confined to suggestions (15) or limited cases for fixed order and without exploiting the important characteristics of the B-spline curve (21).

**Least-squares Fitting of B-spline Curves**. If the defining equation for a B-spline curve

$$\bar{P}(t) = \sum_{i=0}^{n} \bar{B}_i N_{i,k}(t)$$

is considered as yielding a series of algebraic (vector) equations for $m$ known $P(t)$ corresponding to $m$ known data points then the $B_i$ polygon vertices can be obtained by solving the matrix equation

$$[\,S\,]\,[\,B\,] = [\,P\,]$$

where

$$[\,P\,]^T = [\,P(t_0) \quad P(t_1) \quad \cdots \quad P(t_{m-1})\,]$$

$$[\,B\,]^T = [\,B_0 \quad B_1 \quad \cdots \quad B_n\,]$$

and

$$[\,S\,] = \begin{bmatrix} N_{0,k}(t_0) & N_{1,k}(t_0) & \cdots & N_{n,k}(t_0) \\ & & \vdots & \\ N_{0,k}(t_{m-1}) & N_{1,k}(t_{m-1}) & \cdots & N_{n,k}(t_{m-1}) \end{bmatrix}$$

Because in general $[\,S\,]$ will not be square but rather of dimension $m \times (n+l)$ the equation cannot be solved explicitly for $[\,B\,]$. However, a least squares solution of the form

$$[\,B\,] = \left[\,[\,S\,]^T\,[\,S\,]^T\,\right]^{-1}\,[\,S\,]^T\,[\,P\,]$$

can be obtained provided that the parametric values, $t$, for the known data points can be calculated in a reasonable manner. One method, which has been used reasonably successfully, is to approximate the $t$s based on the chord lengths between data points and the maximum parameter value, i.e.,

$$\frac{t_i}{t_{\max}} = \frac{\sum_{j=1}^{i}[P_j(t_j) - P_{j-1}(t_{j-1})]}{\sum_{j=0}^{m}[P_u(t_i) - Pj - 1(t_{j-1})]}$$

**Least-Squares Fitting of B-spline Surface**. Again if the defining equation for a Cartesian product B-spline surface

$$\bar{Q}(u,w) = \sum_{i=0}^{n}\sum_{j=0}^{m} \bar{B}_{i+1,j+1} N_{i,k}(u) M_{j,\ell}(w)$$

is considered as yielding a series of algebraic (vector) equations for $p$ known $Q(u,w)$ corresponding to the $p$ known data points then the $B_{i+l,j+1}$ defining polygon nodes can be ottained by again solving the matrix equation

$$[\,S\,]\,[\,B\,] = [\,Q\,]$$

where now

$$[\,Q\,]^T = [\,Q(u_0,w_0) \quad \cdots \quad Q(u_{p-l},w_{p-1})\,]$$

$$[\,B\,] = [\,B_{1,1} \quad \cdots \quad B_{m+l,n+l}\,]$$

and

$$[\,S\,] = \begin{bmatrix} N_{0,k}(u_0)M_{1,\ell}(w_0) & \cdots & N_{m,k}(u_0)M_{n,\ell}(w_0) \\ & \vdots & \\ N_{0,k}(u_p)M_{0,\ell}(w_p) & \cdots & N_{m,k}(u_p)M_{n,\ell}(w_p) \end{bmatrix}$$

Again, because in general $[\,S\,]$ will not be square but rather of dimension $p \times (m+l)(n+l)$, a least-squares solution of the form

$$[\,B\,] = \left[\,[\,S\,]^T\,[\,S\,]^T\,\right]^{-1}[\,S\,]^T\,[\,P\,]$$

is required. Again the principal difficulty is in effectively determining the parameter values $u_i$, $w_i$ which correspond to the known data points on the surface.

In both of the above discussions a rather large matrix may need to be inverted. However, in contrast to, for example, using cubic splines[1] it need only be inverted once. All subsequent modifications to the curve or surface are made with the polygon vertex control handles. This is a straightforward computational task.

[1]There is strong evidence that nonnormalized cubic splines are preferred for fitting or fairing applications.

## Comparison of Curve Generation Techniques

**Theoretical**. The theoretical characteristics of the various methods of curve generation discussed in this paper are given in Table 1. The particular characteristics discussed are the calculation difficulty, storage requirements, fairness, local control capabilities and available shape control parameters or handles, and the ability to represent a knuckle.

*Parabolic blending* imposes the least computational requirement. The computational technique involves successively generating the $[1 \times 4]$ $t$ parameter matrix for values of the parameter $0 \leq t < 1$, postmultiplying by the constant $[4 \times 4]$ $[A]$ matrix followed by postmultiplying by a [4×3] three dimensional position vector matrix obtained by successively selecting four points from the data. This is quite rapid and is straightforward. Either the first and last spans for the data must be parabolas or two pseudo data points must be added to the data, one at the beginning and one at the end.

**Table 1**. Theoretical characteristics of curve generation techniques.

| | Parabolic Blending | Cubic Spline | Bézier | B-spline |
|---|---|---|---|---|
| Calculation difficulty | Matrix multiplication successive $[1 \times 4][4 \times 4]$ $[4 \times 3]$ Low | Matrix inversion $[m \times m][m \times 3]$† High | Generate basis function Medium-low | Generate knot vector and basis function Medium-high |
| Storage | All $m$ data points plus 2 pseudo end points | All data points | Polygon points | Order plus polygon points |
| Fairness | C1 | C2 | Number of polygon points less $2 - C(m-2)$ | Order less 2 $C(k-2)$ |
| Local control | Good | Poor | Poor | Excellent |
| Shape control parameters | Location of data points | Location of data points End tangent vectors | Location and number of polygon points | Location and number of polygon points Order Location and number of multiple vertices at polygon points |
| Ability to represent knuckle | Must use pseudo off curve points and split curve | Must split curve | Must split curve | $k-1$ repeating or multiple vertices at polygon points |
| Comments | Either first and last span are parabolic or must use pseudo end points | Non-normalized parameters give better fairing | | Control is most effective when used interactively |

†$m$ = number of data or polygon points as approprite

*Bézier curve.* A Bézier curve requires only slightly more computation. The procedure is to generate the basis function as an $[l \times m]$ matrix and postmultiply by the polygon points considered as a $[1 \times 3]$ matrix to yield a point on the curve for each successive value of the parameter $0 \leq t \leq 1$. An example of an algorithm is given in Ref. 1, Appendix C.

*B-spline curve.* Calculation of B-spline curves is of somewhat greater difficulty. Although the calculation of the knot vector is straightforward it involves a number of branches or tests to account for multiple vertices at a given polygon point. The basis function is then generated as an $[m + k \times k]$ matrix for successive values of the parameter $0 \leq t \leq a + k - 2$. The $k$th column is extracted as an $[m \times l]$ matrix, its transpose taken to yield an $[l \times m]$ row matrix which is postmultiplied by the $[m \times 3]$ position vector matrix formed from the polygon points. Alternately a series addition can be accomplished "on the fly" as the basis function is genererated. An example of the latter technique is shown in Ref. 1, Appendix C.

*B-spline curve fit.* When B-splines are used to fit existing data the polygon points required to generate the curve must be found. To accomplish this the $[m \times (n + l)]$ $[S]$ matrix must be generated, premultiplied by its transpose and the inverse of the resulting square product taken. The resulting $[(n + 1) \times (n + l)]$ matrix is postmultiplied by $[S]^T$ and the $[m \times 3]$ data point matrix to yield the required polygon points. The procedure discussed above is then performed to generate the required curve. Provided $(n+1) < m$ the size of the matrix to be inverted is less than that for the cubic spline technique. However, the matrix to be inverted is poorly conditioned for large matrices. Thus, highly accurate or special inversion techniques must be used. The degree of computational difficulty for B-spline fitted curves increases as a function of the increasing order and number of defining polygon points.

*Cubic spline curves.* The generation of a cubic spline curve requires the inversion of an $[m \times m]$ matrix followed by postmultiplication by the $[m \times 3]$ data point matrix. The computational difficulty increases with increasing numbers of data points.

Storage requirements are greatest for cubic splines and parabolic blending. Because in both cases the curves are generated through all the data points, all the data points must be stored. For parabolic blending two additional pseudo points must also be stored. This requires the storage of either $2m$ (2-D) or $3m$ (3-D) floating point or integer numbers for cubic splines and either $2(m+2)$ or $3(m+2)$ for parabolic blending. Because Bézier curves are of fixed order and any number of points along a curve can be generated with a given set of polygon points, only the storage of the polygon points will be required for even very complex curves. Hence, only $3m$ floating point or integer numbers need be stored. B-spline curves require storage of only one additional integer number for the order. Thus, Bézier and B-spline curves potentially represent significant reductions in storage requirements. This can be important for large design projects involving large data bases.

Parabolic blending and cubic spline curves are of fixed C2 (continuity of the second derivative) continuity. The continuity of Bézier curves is fixed by the number of polygon vertices. It is always two less than the number of polygon points. For B-spline curves the degree of continuity is independent of the number of polygon points, it is always two less than the order.

Local control is poor with both cubic splines and Bézier curves. In both cases the entire curve is affected by a change in the location of one of the defining points. Parabolically

blended curves have better local control. A change in the location of one point will affect the curve over four spans, two on either side. A change in the location of one of the pseudo end points will affect the curve only over the first or last span. Local control is excellent for B-spline curves. There are three methods which may be used for local control of B-spline curves. These are: changing the location and number of polygon points, changing the order and changing the number and location of multiple vertices at the polygon points. These effects have been previously discussed and illustrated in Figs. 2 to 7.

For Bézier curves the location and number of polygon points can be changed to achieve shape control. However, the most useful technique is to split the curve. Only the location of the data points or the end tangent vectors can be used to control the shape of either cubic spline or parabolically blended curves whereas the number and location of the polygon points, order and the number and location of multiple vertices can be used for shape control with B-spline curves.

Only a B-spline curve can represent a curve with a knuckle as a single formula. This is accomplished by using $(k-1)$ multiple vertices at a defining polygon point. All other methods considered must split the curve.

**Practical**. In order to determine the practical aspects of the application of these curve generation techniques to ship hull definition four body or station lines representative of various hull forms were selected. These were for a bulbous bow on a destroyer hull, a tugboat with tunnel stern, a trawler with a knuckle at the deck line, and a sailing yacht. These are shown in Fig. 8. In each case, the data were taken from actual designer drawings using an accurate digitizer. Two sets of data were taken for each line. A very complete set with data taken at small intervals along each line. This is shown on the left side in Fig. 8. It was used as a reference for determining the acceptability of the various curve generation techniques. The second set of data consisted of each end point plus a data point at each waterline shown on the original drawing. This is shown on the right in Fig. 8. These data were used to generate "fair" curves with the various techniques. Curves were acceptable when the "designer" was satisfied. Although first and second derivative curves could have been generated and used to evaluate "fairness" this was not done. For each technique, curves were generated using an automatic or batch type program. In addition, for the B-spline technique an initial set of five polygon points was obtained using the automatic programs. The reference curve and these polygon points were then transferred to an interactive graphics system and further fitting undertaken using the CAMILL program discussed below. The results are shown in Table 2 and Figs. 9–12. The B-spline curves were generated using fourth order (third degree) because this yields C2 continuity as does the cubic spline while parabolic blending yields C1 continuity. Although some data storage compression occurs when used in an automatic or batch mode Table 2 shows that the theoretical advantages of B-spline curves are in general not realized in practice. This is a result of the inaccurate representation of the parameter values for the data points when used to generate the $[S]$ matrix discussed above. The scheme uses a chord approximation to the curve length. As the number of polygon points is reduced to achieve data storage compression the approximation worsens. Increasing the number of polygon points above about 10 or 11 requires the inversion of a large (edge size 10 or 11) poorly conditioned matrix. Under these conditions standard single precision matrix inversion techniques do not always yield acceptable results. Except for the

yacht line it was necessary to split the curves in order to obtain acceptable results. Higher order B-spline fits were attempted for the bulbous bow. The results were unacceptable due to typical spurious "wiggles" experienced with higher order polynomial curve fits.

Table 2 shows that the interactive curve fit was much more successful. In all cases a single representation of the curve was obtained. This includes the two cases which exhibit a "knuckle". Except for part of the trawler curve a fourth order B-spline was used. Greatest data compression occurred for the bulbous bow where 6 polygon points vice 26 data points were used. When the B-spline curve generation algorithm is coded in straightforward floating point FORTRAN and used with a rubber banding technique, the computational load will make most minicomputer based interactive refresh graphics systems marginally acceptable. Because of this, each interactive curve fit took approximately twenty minutes. Faster algorithms coded using either integer arithmetic FORTRAN or assembly language eliminates this effect. For interactive refresh graphics the least computational load is imposed by the parabolic blending technique.

## Numerically Controlled Milling of Ship Hulls - CAMILL

Having investigated the various curve generation techniques for ship hull definition it still remains to utilize them in a practical computer aided design system with a practical end product. This was accomplished through development of a system called CAMILL (Computer Aided Milling). CAMILL is intended to be a simple approach to the problems of combining interactive graphics and numerically controlled (NC) milling. The emphasis of CAMILL is ship hull design and the automatic milling of a ship hull model.

The major hardware components of CAMILL are an interactive refresh graphics system and a numerically controlled three axis milling machine. The graphics system is an Evans and Sutherland Computer Corporation PICTURE SYSTEM, driven by a Digital Equipment Corporation PDP-11/45. User interaction with the graphics system is primarily through the use of function switches, control dials and a tablet. The actual milling of the ship hulls is performed with a Pratt and Whitney TRIMAC XV computerized numerically controlled three axis milling machine. To supplement the milling machine control panel and act as a front end processor, a Tektronix 4051 Graphic System has been interfaced to the TRIMAC minicomputer controller. For obtaining high resolution hard copy plots at various stages of the design process, there is a large (4'x8') ZYNETICS flat bed plotter which is controlled off line by magnetic tape.

The two major software components of CAMILL consist of an interactive display program and a cutting program. The graphics program uses the Picture System to allow the creation and manipulation of lines which represent the ship hull design. When the design is ready for milling, a data file is generated which is transferred by magnetic tape to the cutting program. The cutting program is executed in the Tektronix 4051 and directs the milling process through the TRIMAC controller. This "on-line cutting" is a variation from the traditional approach, which is to produce a paper tape from an APT-post processor combination which is then read by the NC machine controller to perform the milling.

Body lines and their spacing along the ship's longitudinal axis are assumed to be fundamental to the design. This allows waterlines and buttock lines to be automatically generated

from a body plan. The display program also has the capability of generating a three dimensional view of the hull which may be either perspective or orthographic.

The display program provides the designer with capabilities for interactively generating curves using any of the techniques discussed above. The body plan may be entered either as an external file or generated by directly digitizing points from the tablet. Facilities for interactively modifying the lines are provided through the tablet and menu selection. The internal structure of the display program is designed such that any modification to one set of lines is appropriately included in the other sets. User interaction with the three dimensional view is provided through use of the control dials to perform rotation and translation about and along the $X$, $Y$, $Z$ axes. An additional dial is used to control sectioning which allows the display of a thin slice of the three dimensional hull.

As a particular design nears completion, the designer may want to verify the correctness or fairness of the lines. This verification can be accomplished by making a large scale plot. The XYNETICS plotter allows drawings up to 50 inches by 89 inches. Thus, to verify a set of lines, the designer need only select a set of lines and initiate the creation of a plot file. The plot file is later transferred to magnetic tape and plotted at the desired scale. By plotting the lines at a large scale, any irregularities in the smoothness or fairness of the lines will be much more apparent than when viewed on the refresh display.

When the design is complete, the display program is directed to create a "milling file". The milling file is essentially the path to be followed by the cutting head of the NC milling machine. The milling file is transferred by a magnetic tape cassette to the Tektronix 4051 processor which directs the actual cutting. The contents of the milling file is primarily water lines. These lines provide a straightforward technique for milling the ship hull. Due to limitations in memory size of the 4051 and the sequential nature of the 4051 cassette tape the display program must perform two functions in creating the milling file. The first function is to sort the water lines according to depth. Because the accuracy of the finished product is determined by the number of water lines in the milling file, the second function is to generate enough intermediate water lines to produce the desired accuracy. The hull is then cut upside down.

The TRIMAC milling machine and the Tektronix 4051 are the two main hardware components that execute the cutting program. The cutting program has four major functions. First, it acts as a "Post Processor" by converting the milling file into TRIMAC machine commands. Second, the cutting program uses the machine commands to actually direct the cutting of the model hull. Third, the cutting program assists the mill operator in machine setup and provides him with additional control during the milling process. Finally, the cutting program solves any potential interferences between the model and the machine tool. To date the CAMILL program has been successfully used to produce three small (2-4') models. Additional details on CAMILL are given in Ref. (22).

## Conclusions

The advantages and disadvantages of parabolic blending, cubic splines and B-spline curve generation techniques have been investigated in the context of ship hull definition. The results show that no one curve generation technique is a panacea. B-spline curves do not

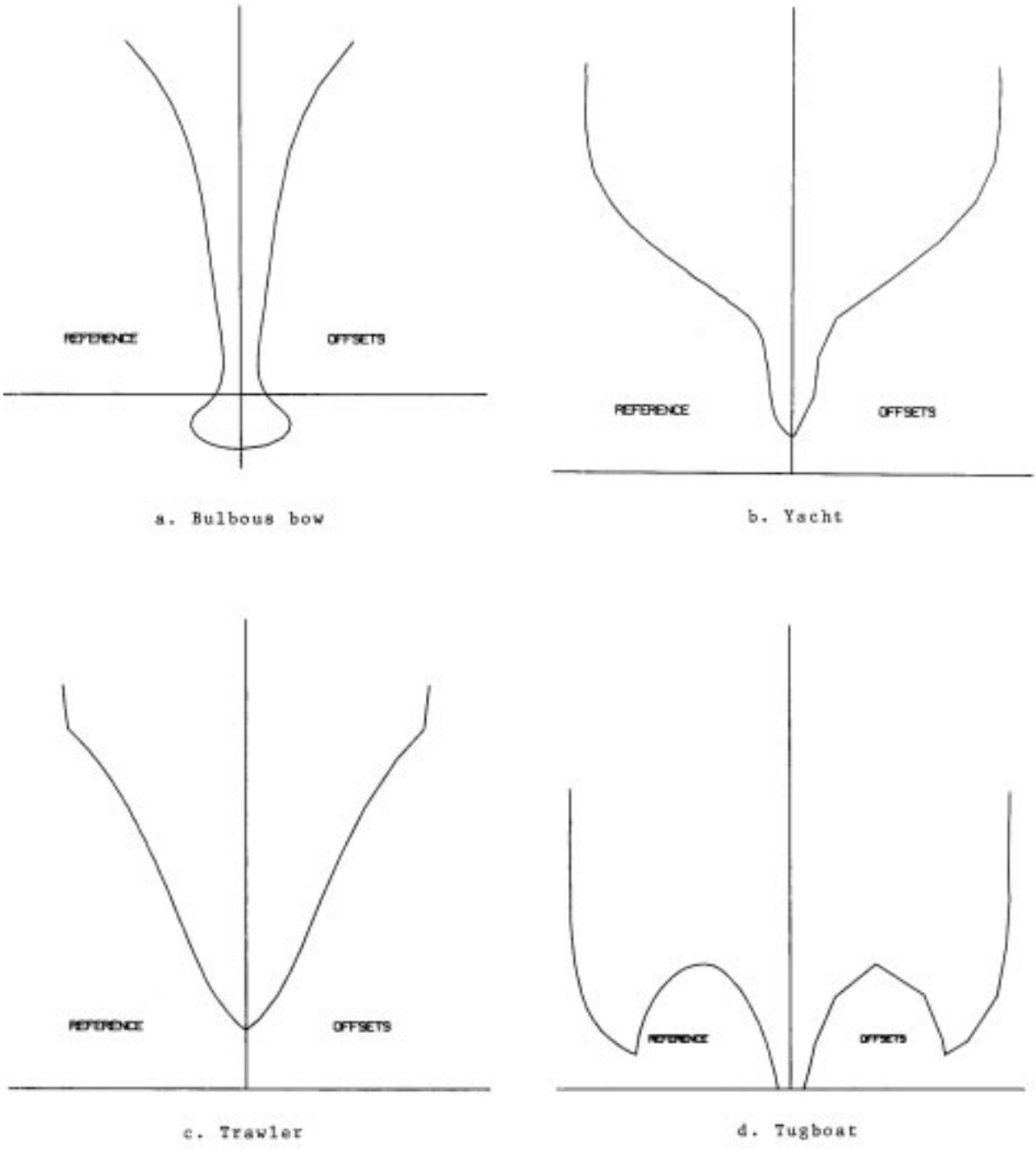


**Figure 8.** Various hull station (body) lines.

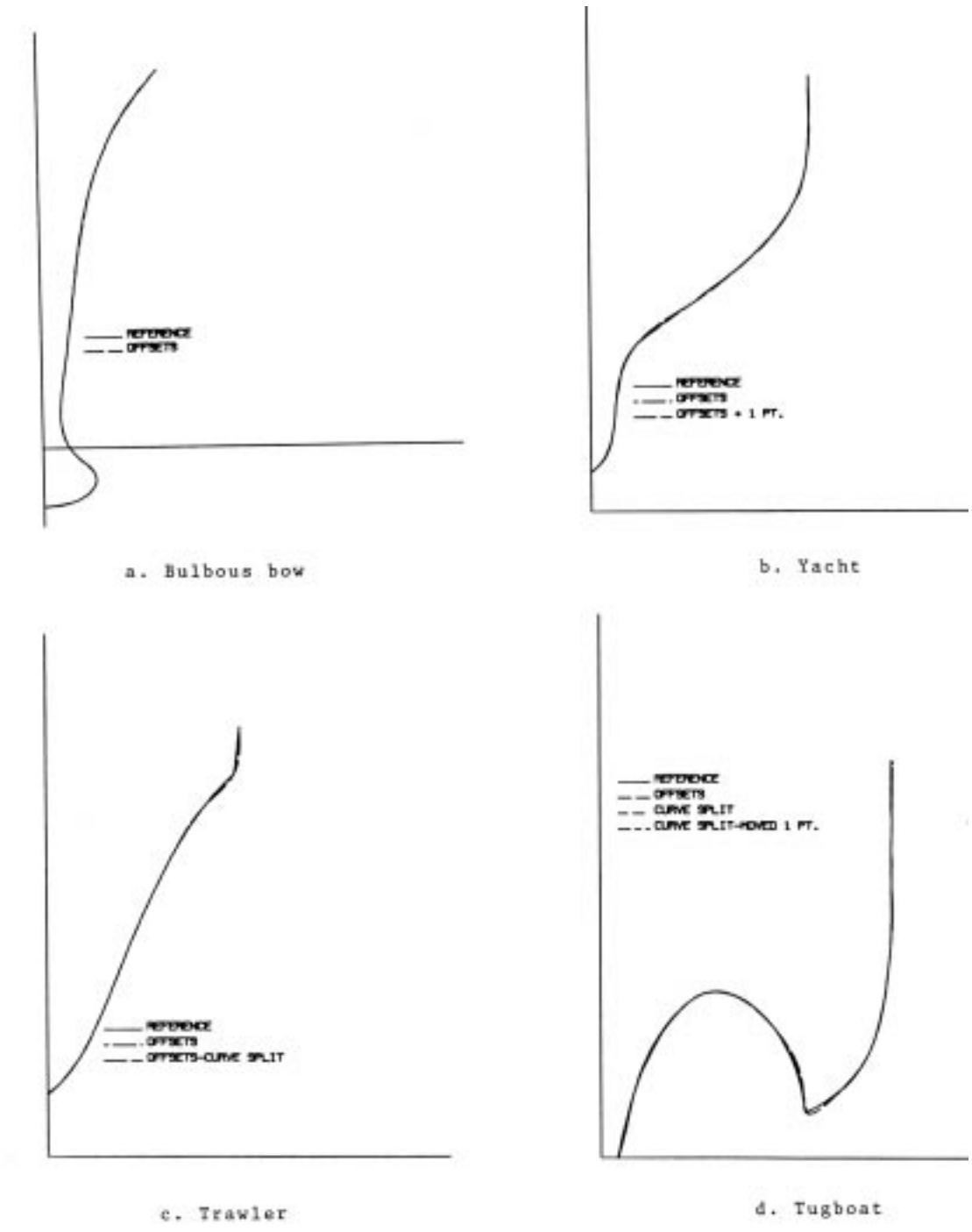


**Figure 9.** Cubic spline results.

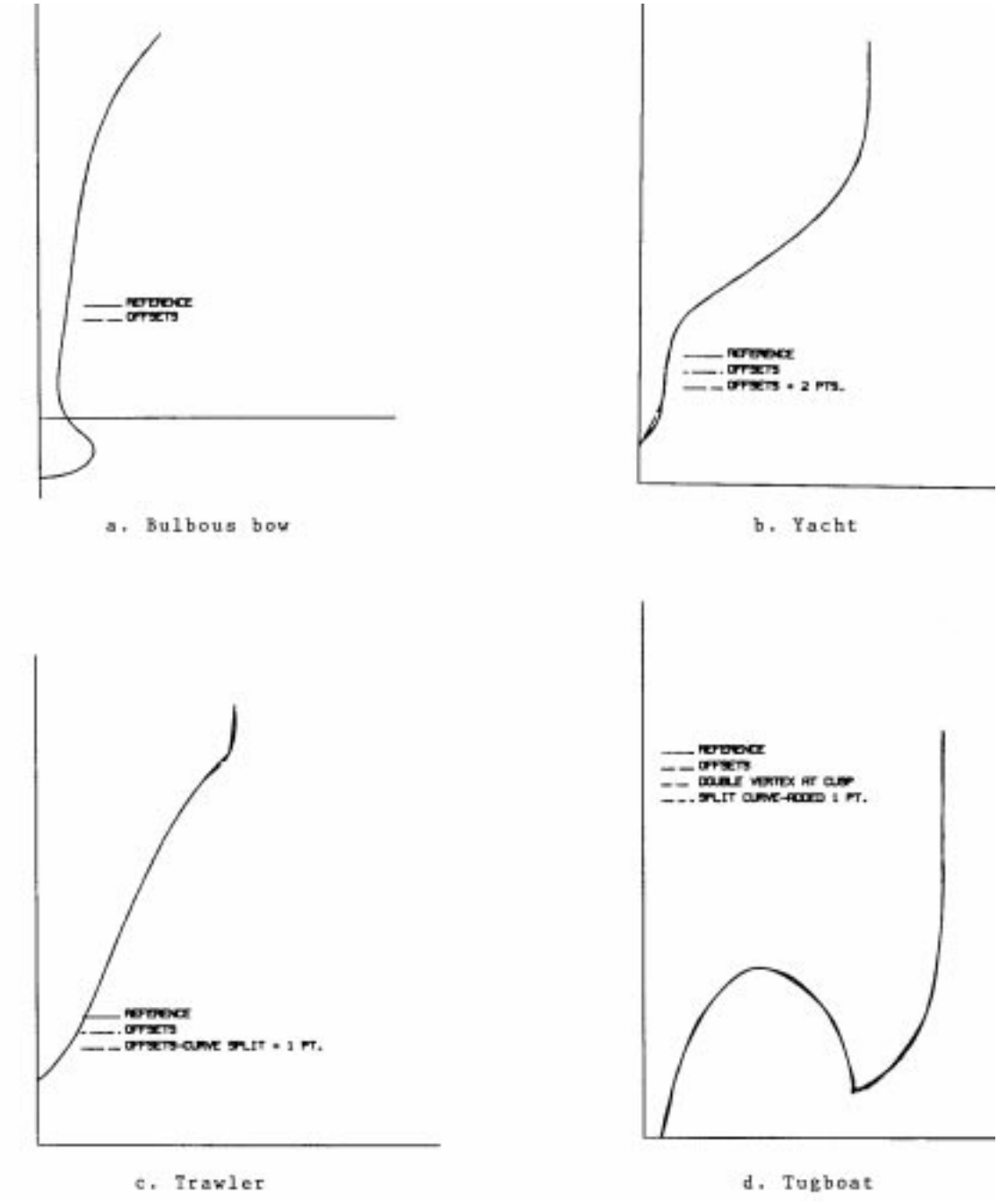


**Figure 10.** Parabolic blending results.

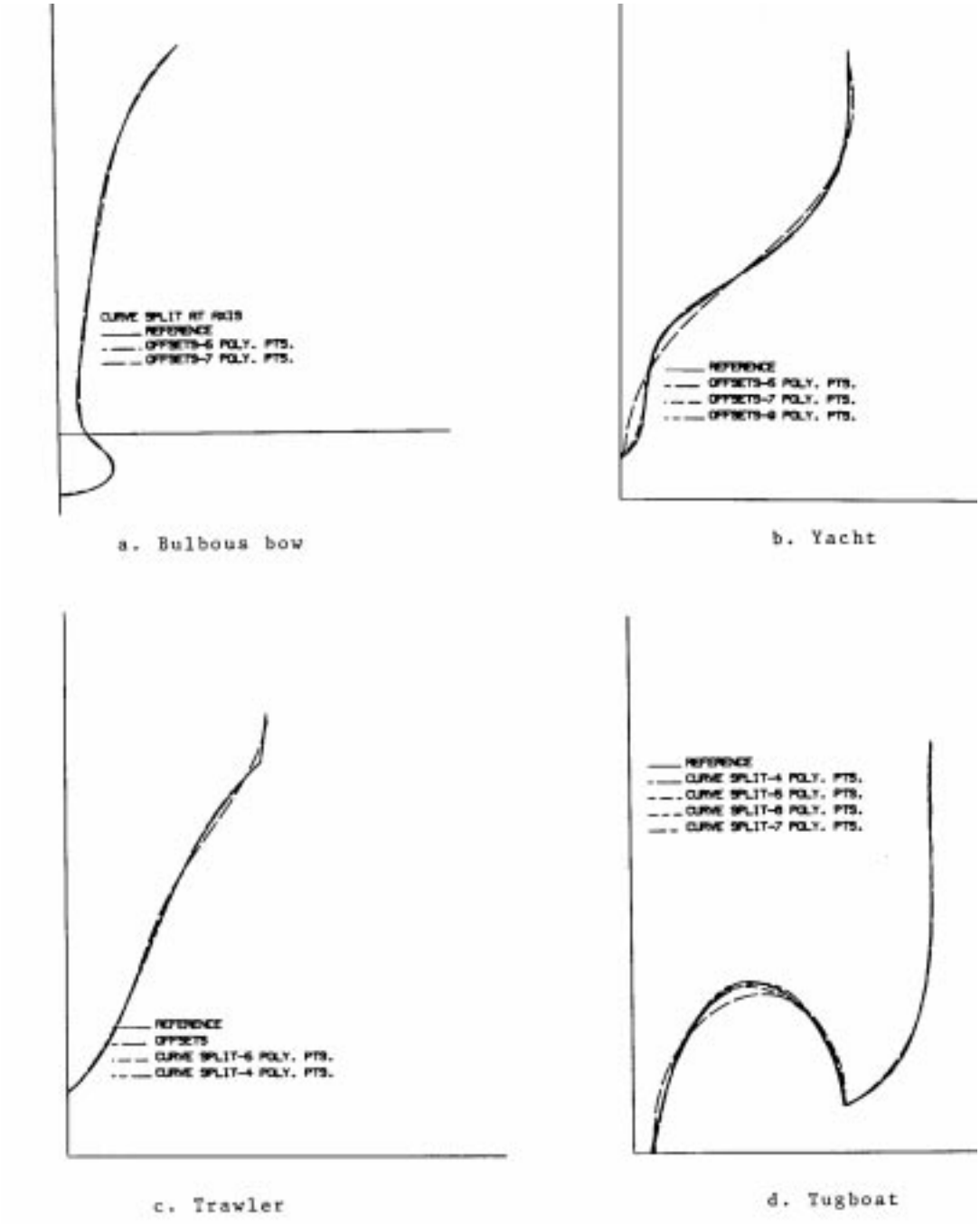


**Figure 11.** Automatic B-spline fitting.

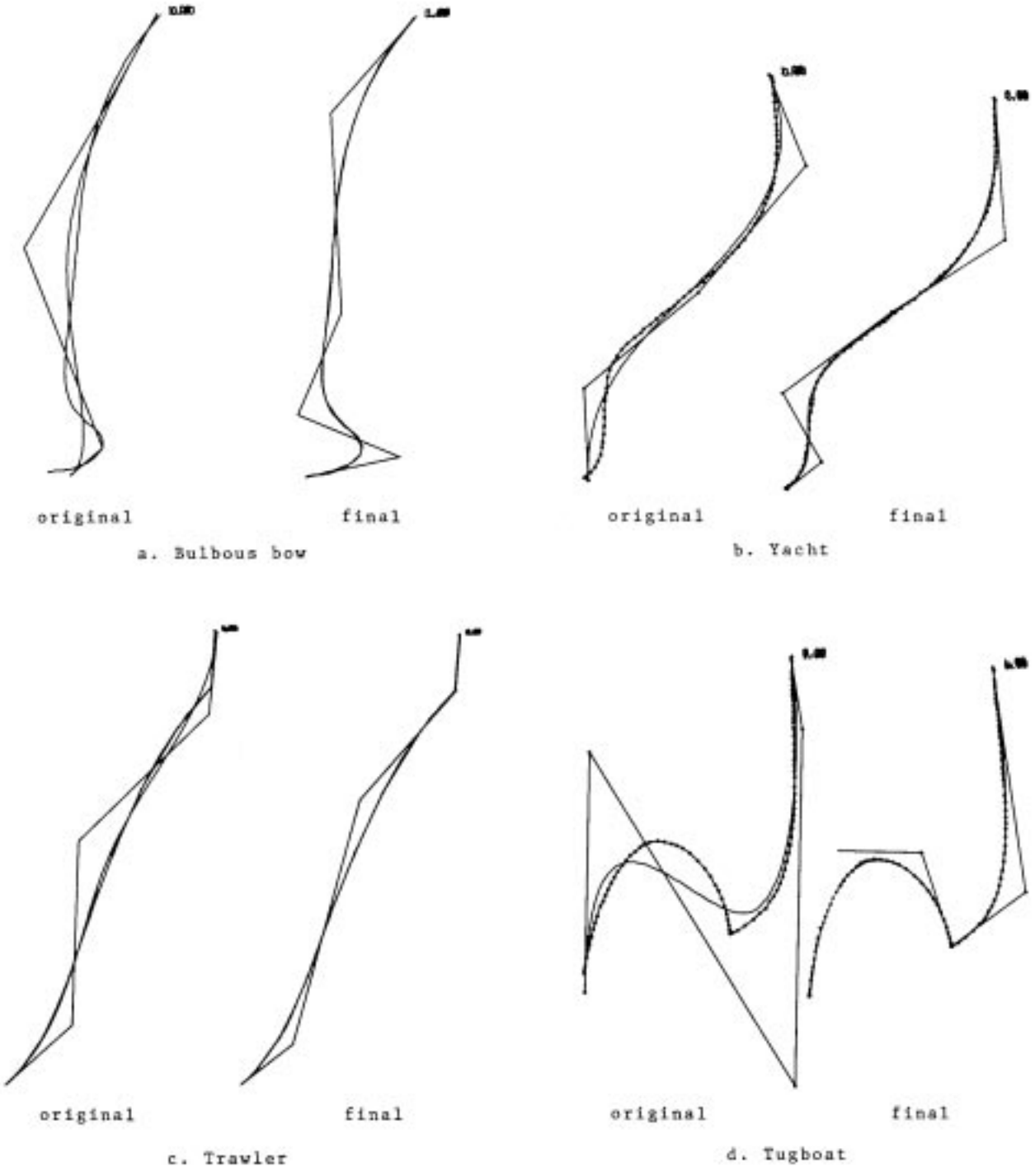


**Figure 12.** Interactive B-spline fitting.

exhibit the anticipated large reduction in data storage when used in an automatic or batch mode. When used interactively the anticipated data storage reductions are easily achieved. The investigated curve generation techniques have been incorporated into an interactive computer aided ship design system. This system has been successfully integrated with a numerically controlled milling machine through a Tektronix 4051 graphics system. Ship models have been successfully cut.

## Acknowledgments

The work of Steven Satterfield in implementing the interactive graphics system and of Francisco (Paco) Rodriquez in implementing the numerically controlled cutting program is sincerely acknowledged.

## References

1. D. F. Rogers and J. A. Adams, Mathematical Elements for Computer Graphics, McGrawHill, New York, 1976.
2. N. E. South and J. P. Kelly, "Analytic Surface Methods," Ford Motor Company N/C Development Unit, Product Engineering Office, December 1965.
3. A. A. Nutbourne, "A Cubic Spline Package Part 2 - The Mathematics," Computer Aided Design, Vol. 5, No. 1, January 1973.
4. J. Alan Adams, "A Comparison of Methods for Cubic Spline Curve Fitting," Computer Aided Design, Vol. 6, pp. 1-9, 1974.
5. Denman, H. H. "Smooth Cubic Spline Interpolation Functions," Industrial Mathematics, J. Ind. Math. Soc., Vol. 21, Part 2, pp. 55-75, 1971.
6. Cline, "Curve Fitting Using Splines Under Tension," Atmos. Tech., No. 3, pp. 60-65, 1973.
7. D. C/ Schweikert, "An Interpolation Curve Using a Spline in Tension," J. Math. Phys., Vol. 45, pp. 312-317, 1966.
8. A. W. Overhauser, "Analytic Definition of Curves and Surfaces by Parabolic Blending," Technical Report No. SL68-40, Ford Motor Company Scientific Laboratory, May 8, 1968.
9. Bézier, Emploi des Machines a Commande Numerique, Masson et Cie, Paris, 1970. Translated by D. R. Forrest and A. A. Pankhurst, as P. F. Bézier, Numerical Control, Mathematics and Applications, John Wiley and Sons, Inc., London, 1972.
10. E. Mehlum, "Curve and Surface Fitting Based on VAriational Criteriae for Smoothness," Central Institute for Industrial Research (CIIR), Oslo, Norway, December 1969.
11. Autokon Systems.
12. CASDOS private communication.

13. P. E. Bézier, "Example of an Existing System in the Motor Industry: The Unisurf System," Proc. Roy. Soc. (London) Vol. A321, pp. 207-218, 1971.

14. R. A. Riesenfeld, "Berstein-Bézier Methods for the ComputerAided Design of Free-Form Curves and Surfaces." Ph.D. Thesis, Syracuse University, March 1973.

15. J. Brewer and D. Anderson, "Visual Interaction with Overhauser Curves and Surfaces," SIGGRAPH 77 Fourth Annual Conference on Computer Graphics and Interactive Techniques, 20-22 July 1977, San Jose, California.

16. A. R. Forrest, "Interpolation and Approximation by Bézier Polynomials," CAD Group Doc. No. 45, UML, Cambridge University, October 1970.

17. I. J. Schoenberg, "Contributions to the Problem of Approximation of Equidistant Data by Analytic Functions," Q. Appl. Math., Vol. 4, 1946, pp. 45-99; 112-141.

18. M. G. Cox, "The Numerical Evaluation of B-Splines," National Physical Laboratory DNAC 4, August 1971.

19. Carl de Boor, "On Calculating with B-Splines," J. Approx. Theory, Vol. 6, pp. 50-62, 1972.

20. R. E. Barnhill and R. F. Riesenfeld, Computer Aided Geometric Design, Academic Press, New York, 1974.

21. J. McKee, Private communication.

22. Steven G. Satterfield, Francisco Rodriquez, and David F. Rogers, "A Simple Approach to Computer Aided Milling with Interactive Graphics," presented at SIGGRAPH '77 Fourth Annual Conference on Computer Graphics and Interactive Techniques, San Jose, CA, July 20-22, 1977.